Decsion Tree

In [1]:
```
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from pandas.tools.plotting import parallel_coordinates, andrews_curves
import matplotlib.pyplot as plt
import pydotplus
import urllib.request

%matplotlib inline
```

In [3]:
```
import numpy as np
import pandas as pd
names = ['Company','Open','High','Low','Close','Volume']
data = pd.read_csv('stockpredictions.csv', names=names)
print(data.shape)
```

```
(513, 6)
```

In [4]:
```
data.sample()
```

Out[4]:

	Company	Open	High	Low	Close	Volume
113	ABC	23.75	23.855	23.53	23.81	23858

In [5]:
```
data.head()
```

Out[5]:

	Company	Open	High	Low	Close	Volume
0	AA	11.32	11.34	10.87	11.00	585630
1	AA	11.01	11.11	10.82	10.91	258446
2	AA	10.95	10.95	10.61	10.84	233940
3	AA	10.72	10.75	10.41	10.41	282508
4	AA	10.50	10.68	10.31	10.58	267170

In [6]:
```
data.tail()
```

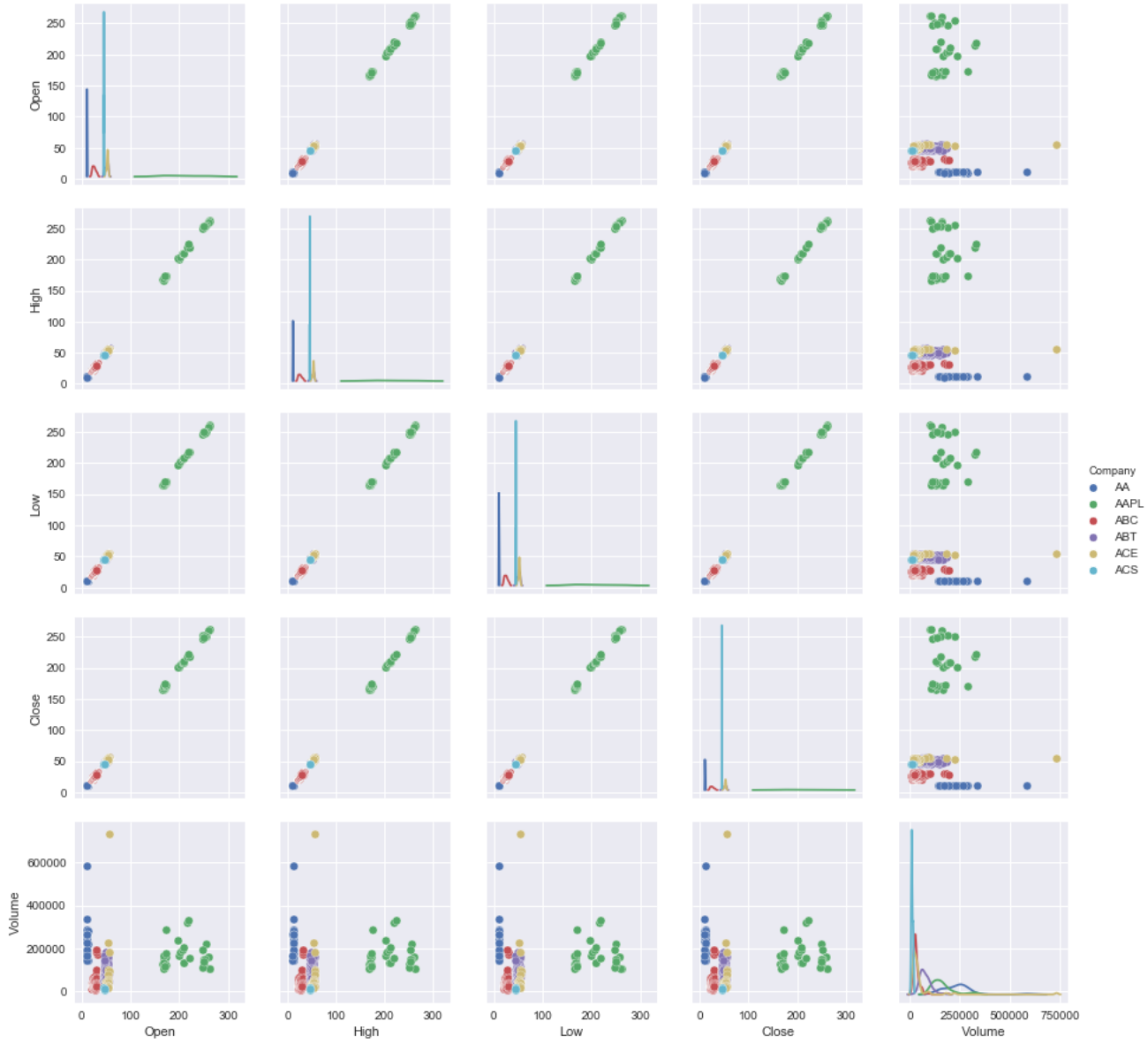
Out[6]:

	Company	Open	High	Low	Close	Volume
508	ACE	52.91	53.545	52.91	53.51	17228
509	ACS	45.08	45.330	44.40	45.32	17241
510	ACS	45.55	45.950	45.40	45.60	9765
511	ACS	45.87	45.930	45.45	45.49	8212
512	ACS	45.53	45.880	45.36	45.49	11869

In [9]:
```
sns.pairplot(data, hue = 'Company', diag_kind="kde")
```

```
C:\Users\Jatan\Anaconda3\lib\site-packages\statsmodels\nonparametric\kdetools.py:20:
VisibleDeprecationWarning: using a non-integer number instead of an integer will res
ult in an error in the future
  y = X[:m/2+1] + np.r_[0,X[m/2+1:],0]*1j
```

Out[9]: <seaborn.axisgrid.PairGrid at 0x8383e05a20>



In [10]:
```
encoder = LabelEncoder()
data['Company'] = encoder.fit_transform(data['Company'])
```

In [11]:
```
data.sample()
```

Out[11]:

	Company	Open	High	Low	Close	Volume
199	2	29.82	30.085	29.65	29.97	28542

In [12]:
```
data.head()
```

Out[12]:

	Company	Open	High	Low	Close	Volume
0	0	11.32	11.34	10.87	11.00	585630
1	0	11.01	11.11	10.82	10.91	258446
2	0	10.95	10.95	10.61	10.84	233940
3	0	10.72	10.75	10.41	10.41	282508
4	0	10.50	10.68	10.31	10.58	267170

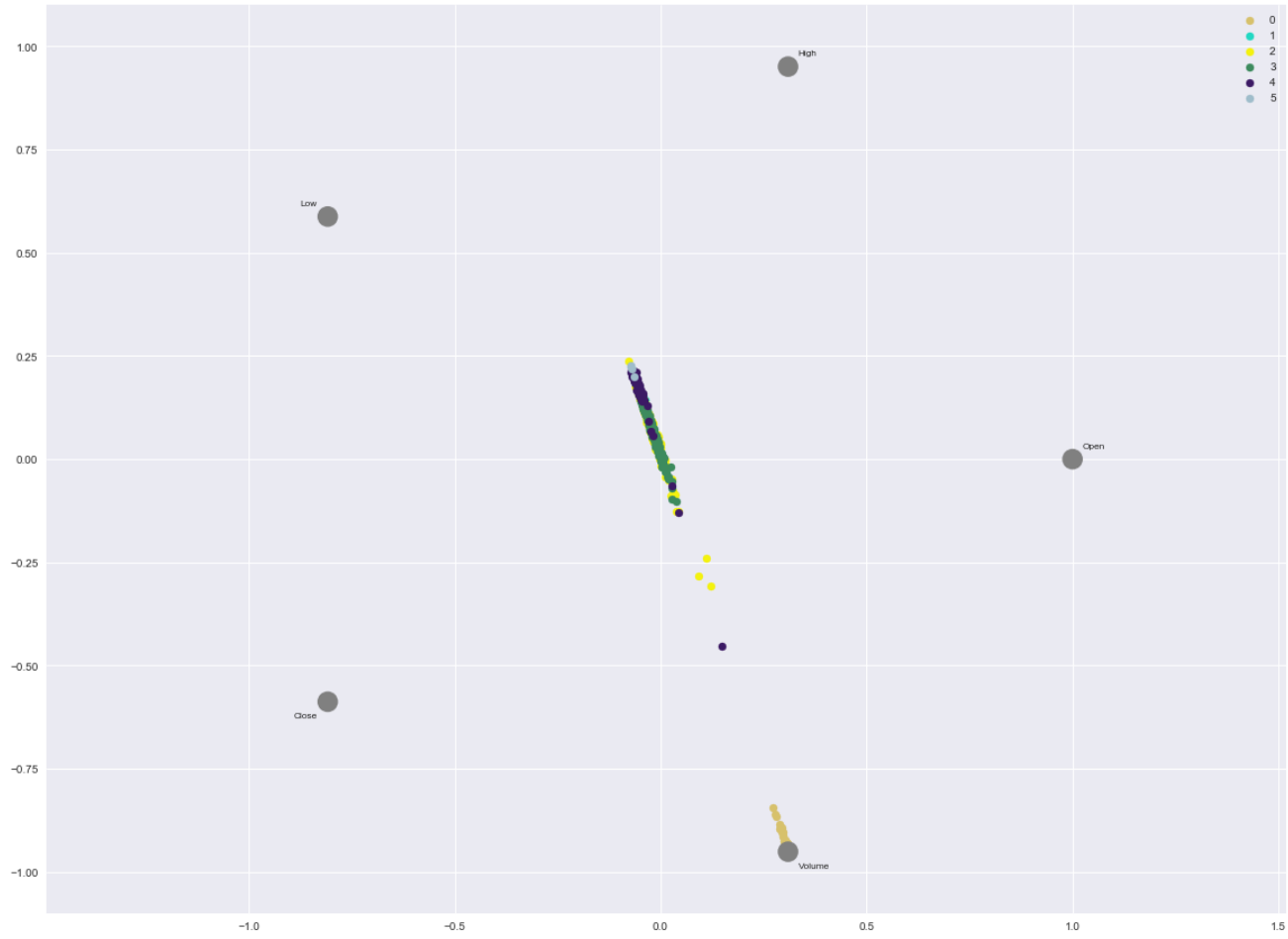
In [13]:
```
data.tail()
```

Out[13]:

	Company	Open	High	Low	Close	Volume
508	4	52.91	53.545	52.91	53.51	17228
509	5	45.08	45.330	44.40	45.32	17241
510	5	45.55	45.950	45.40	45.60	9765
511	5	45.87	45.930	45.45	45.49	8212
512	5	45.53	45.880	45.36	45.49	11869

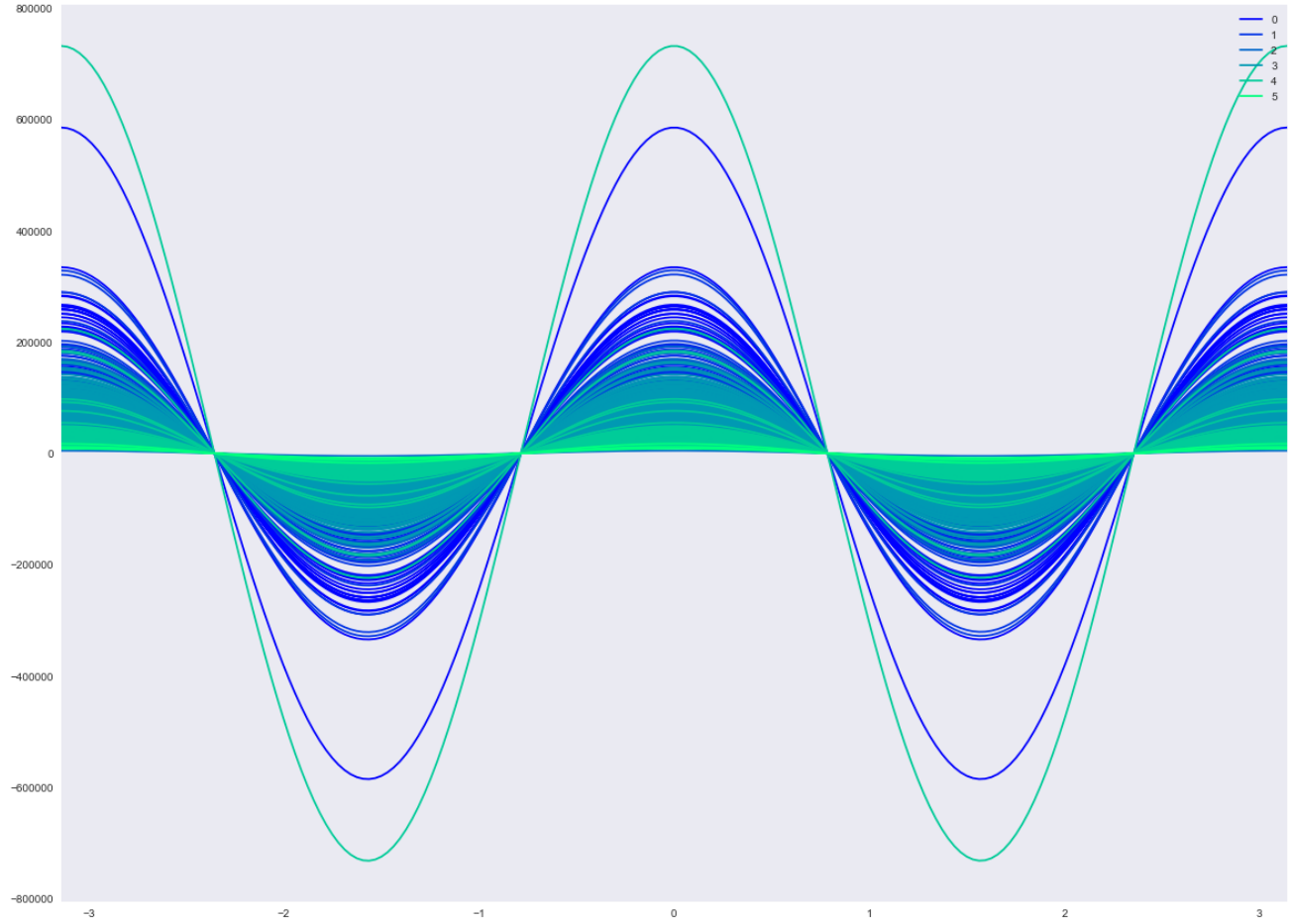
In [14]:
```
width, height = 20, 15
size = width, height
plt.figure(figsize = size)
from pandas.tools.plotting import radviz
radviz(data, 'Company')
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x8393b2b9e8>
```



In [15]:
```
plt.figure(figsize = size)
andrews_curves(data, 'Company', colormap = 'winter')
```

Out[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x8394f537f0>
```



In [17]:
```
df = (data - data.min()) / (data.max() - data.min())
df.sample(5)
```

Out[17]:

	Company	Open	High	Low	Close	Volume
52	0.2	0.827393	0.848846	0.829524	0.845906	0.445127
271	0.6	0.177028	0.175304	0.175472	0.175340	0.133780
335	0.6	0.174200	0.173404	0.175072	0.174704	0.094223
443	0.6	0.160859	0.159785	0.160326	0.161017	0.055207
103	0.4	0.052730	0.051665	0.051950	0.051802	0.022765

In [18]:
```
from sklearn.model_selection import train_test_split
```

In [19]:
```
train, test = train_test_split(data, train_size = 0.60)
```

In [20]:
```
len(train), len(test)
```

Out[20]:
```
(307, 206)
```

In [21]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [22]:
```
features = data.columns[1:]
target = data.columns[0]
```

In [23]:
```
model = DecisionTreeClassifier()
model = model.fit(train[features], train[target])
```

In [24]:
```
from sklearn.metrics import accuracy_score
```

In [25]:
```
predicted = model.predict(test[features])
```

In [26]:
```
print('Accuracy Score:', accuracy_score(test[target], predicted) * 100)
```

```
Accuracy Score: 93.2038834951
```

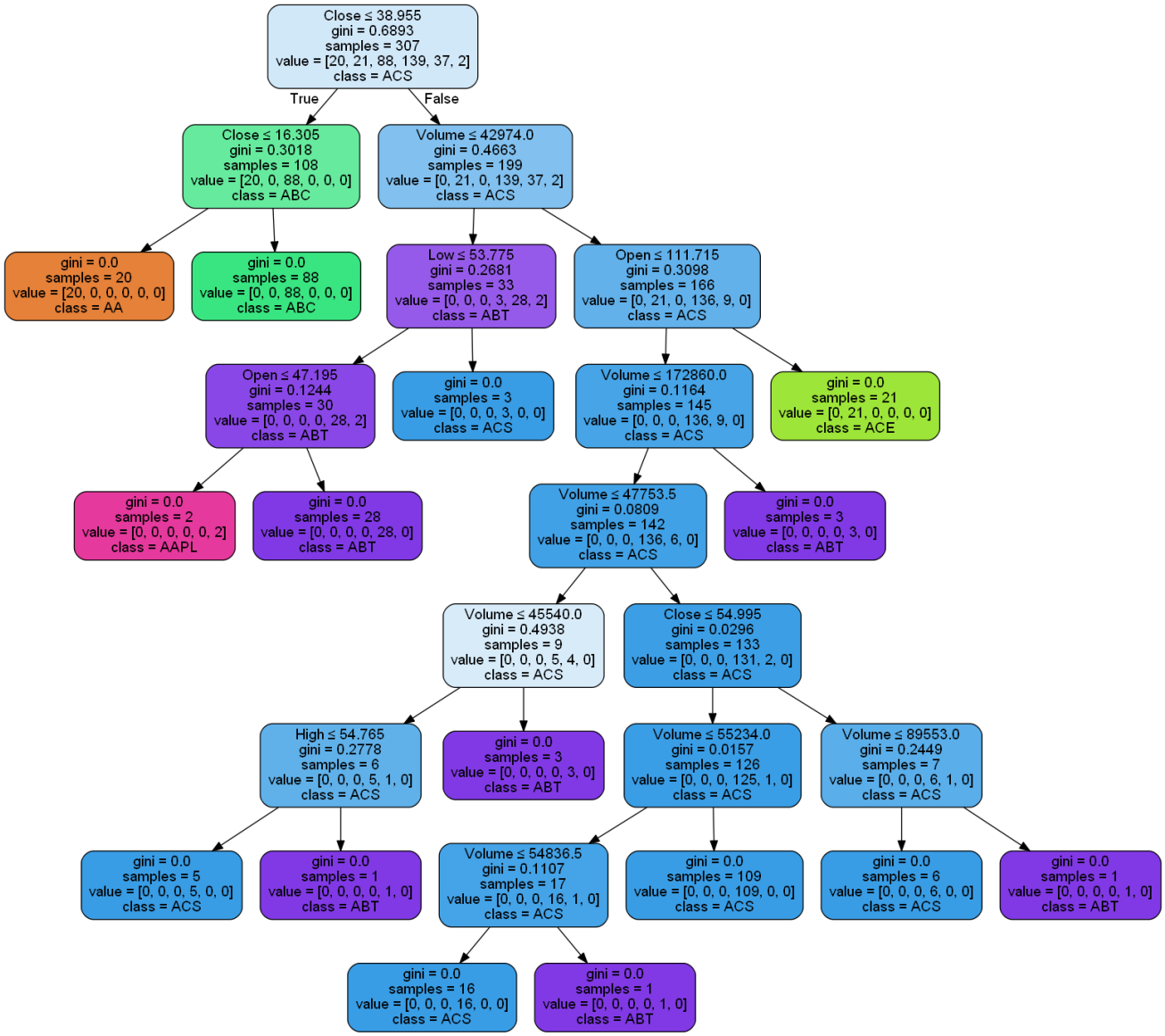
In [27]:
```
from sklearn.tree import export_graphviz
```

In [29]:
```
Class_names = list(set(encoder.inverse_transform(data[target])))
export_graphviz(model,out_file = 'SP.dot',
                        feature_names = features,
                        class_names = Class_names,
                        filled = True,
                        rounded = True,
                        special_characters = True)
graph = pydotplus.graphviz.graph_from_dot_file("SP.dot")
```

In [30]:
```
png = graph.create_png()
```

In [31]:
```
from IPython.display import Image
Image(png)
```

Out[31]:



In [32]:
```
model = DecisionTreeClassifier(criterion = 'entropy')
model = model.fit(train[features],train[target])
```

In [33]:
```
predicted = model.predict(test[features])
```

In [34]:
```
print('Accuracy Score : ', accuracy_score(test[target], predicted)*100)
```

```
Accuracy Score :  94.1747572816
```

In [35]:
```
ddata = export_graphviz(model,
                        out_file = None,
                        feature_names = features,
                        class_names = Class_names,
                        filled = True,
                        rounded = True,
                        special_characters = True)
```

In [36]:
```
graph = pydotplus.graph_from_dot_data(ddata)
```

In [37]:
```
png = graph.create_png()
```

In [38]:
```
Image(png)
```

Out[38]: